DEVELOPMENT GUIDE

def convert_file_to_DNA (path)

This function is to convert a file to original DNA sequence, firstly compressing the file to zip package, secondly encrypting the zip package by ISAAC algorithm and converting it to quaternary, thirdly converting the quaternary to original DNA sequence by 0 to A, 1 to C, 2 to G and 3 to T. It aims to form the original DNA sequence.

def convert_DNA_to_file (path)

This function is to convert a DNA sequence to file, converting DNA to bit and decrypting by isaac algorithm then generating a zip package, at last, decompressing the zip package to get the original file.

def to DNA (string)

All information is added to original sequence and sub sequence is quaternary. We convert the quaternary to DNA by 0 to A, 1 to C, 2 to G and 3 to T firstly. And then we make the DNA random following the rule in table.1. (The first character is unchanged. For the following sequence, characters are taken from the row defined by the previous changed character conversion.)

previous /next	A	C	G	T
A	C	G	T	A
C	G	T	A	C
G	T	A	C	G
T	A	C	G	T

Table.1 Random process

def to NUM (dna)

This function is to record the DNA sequence using following table and then convert the ordered DNA sequence to quaternary by 0 to A, 1 to C, 2 to G and 3 to T. (The first character is unchanged. For the following sequence, characters are taken from the row defined by the previous character conversion.)

previous /next	A	C	G	T
A	T	A	C	G
C	G	T	A	C
G	C	G	T	A
T	A	C	G	T

Table.2 Record DNA Sequence

def encoding_file (request, seq, fname, ftype)

Write the sub sequences into a file according to the file type given by users which includes txt, fasta, and SBOL-Xml.

def reverse_s (string)

This function is to reverse a string and return the reversed string.

check (dna_list, length)

Use fuzzy algorithm to check the DNA sequences in the dna_list and return a correct string.

def file_seq(file)

This function is to get all sequences from a file. It accepts file type including txt, fasta and xml format. The txt file contains pure DNA sequences, and Xml is in the standard of SBOL. Return a list of all DNA sequences.

class encoding (object)

def __init__ (self, path)

This function is the working function, it calls other functions to convert file to final DNA sequences. It runs with the following steps.

1. Add length information to the end of the original DNA sequence and make sure the sequence length as the multiple of 50.
2. Split the DNA sequence to units with 50bp per unit, join four units to a sub sequence and make it four times-fold redundancy. And then, reverse odd units.
3. Add index and check of index information to sub sequences.
4. Add A or T to the head of sub sequences and C or G to the tail of sub sequences to label the order of sub sequences.

def make_S5 (self)

Count the length of the original DNA sequence and turn it to quaternary(S2), add “0”(S3) before S2 to ensure the total length of original DNA sequence, S3 and S2 is the multiples of 50. (The order is original DNA sequence, S3, S2. S2 and S3 are converted to DNA by ‘toDNA()’ function). Return the treated DNA sequence as S5.

def make_F (self)

Split the DNA sequence to units with 50bp per unit, join four units to a sub sequence and make it four times-fold redundancy. And then, reverse odd units. Return a list(F) which contains the primary sub sequences.

def make_subSequence(self)

Add index and check information of index(P) to every sub sequence, then add A or T to the head of sub sequences and C or G to the tail. Return a list which contains the final sub sequences.

class decode (object)

def __init__ (self, path, fname)

This is function is to convert sub sequences to file by the following steps.

1. Get the sequences from a file.
2. Get the index of sub sequences and P, check the index by parity-check. Then, order the sub sequences by analyzing that starting with A or T and ending with C or G.
3. Check the sub sequences which have the same index by fuzzy algorithm and get the correct sub sequence of each index.
4. Split the sub sequences into 50bp per units and check the corresponding units in different sub sequences by fuzzy algorithm and get the correct unit.
5. Connect the units and get the length information of original DNA sequence.
6. Get the original sequence and write it into a text file.
7. Convert the original DNA sequence to bit by nt2bit, then decryption it and convert it to zip package by ISAAC.
8. Decompress the zip package and get the original file.

def get_Fi (sequence)

Get the index of sub sequences and P, check the index by parity-check. Then, order the sub sequences by analyzing that starting with A or T and ending with C or G. Check the subsequences which have the same index by fuzzy algorithm and get the correct sub sequence of each index. Return it as Fi.

def get_seq(F)

Split the sub sequences into 50bp per units and check the corresponding units in different sub sequences by fuzzy algorithm and get the correct unit. Connect the units and get the length information of original DNA sequence. Get the original sequence and write it into a text file.